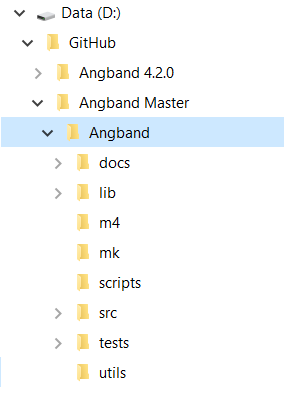
These instructions were developed for building and developing Angband using Visual Studio 2019 Community Edition (the free version of VS). As of the writing of these instructions, Angband 4.2.0 is the supported version, but the ‘master’ version contains additional changes and fixes, some of which are needed to build with VS. Those will be noted below.

1. Install Visual Studio Community 2019 from Microsoft.
   1. I didn’t keep a record of my install experience, so the following is from memory.
   2. I think I chose the “Desktop development with C++” workload
   3. I don’t think I needed or chose any optional components during the install
   4. After VS was installed I ran it. If the top menu bar wasn’t displayed I turned it on.
   5. Under Extensions->Manage Extensions I searched for Git. There are several extensions that support Git, the one I chose was “GitHub Extension for Visual Studio”, created by GitHub, It is version 2.10.8.8132 and was free.
2. Register with GitHub and connect from within VS
   1. It’s easiest to register outside of VS, using a web browser pointed at github.com.
   2. Once you have github credentials and they are validated via email, open VS.
   3. The left hand pane of VS has tabs at the bottom, one of them should be Team Explorer, click on the tab. If the tab isn’t there for you yet, use the menu bar View->Team Explorer to bring it up.
   4. You should see a section named simply “GitHub”. In the GitHub section, click Sign In and enter your GitHub credentials and get yourself signed into GitHub from within VS and connected.
3. Set up a directory structure somewhere on your computer where you will build & develop Angband.
   1. Start with an empty folder somewhere convenient to you. This will be your “solution folder”.
   2. You can set the name of the solution folder to something that denotes not only that it’s for Angband but also which version of Angband. For example, I have one named Angband 4.2.0 and another named “Angband Master” and a third where I’m doing development for my personal variant.
   3. Create a subfolder within your solution folder, and name it Angband. This will be your “project folder” or “project directory”. Notes:
      1. The checked-in VS files have a dependency on the project directory being named Angband, so use exactly that.
      2. For VS development, it’s best to stick with the VS standard organization of project directories being in an immediate sub-folder of the solution folder rather than mixing everything into one folder. Separating them allows you to later create other related projects underneath the same solution, for example for testing or tools or scripts that you use with Angband but aren’t part of the game itself.
      3. This folder must be empty for Git Clone to work.
4. Clone Angband
   1. In VS, in the Team Explorer, in addition to the GitHub section where you logged in, you should also see a “Local Git Repositories” section, with New, Add, Clone and View Options pulldowns. Click the Clone pulldown.
   2. In the (yellow) text box, where it says ‘enter the URL of a git repository’, enter:

<http://github.com/angband/angband> (this clones master, I’m not sure what URL you enter to get old versions)

* 1. In the text box below that, enter the path to your project directory, which is the lower of the two folders you created in step III c, the one that must be empty.
  2. Make sure Recursively Clone Submodules is checked
  3. Click the Clone button and wait for it to download everything. Status shows in the VS Output window.
  4. At this point your solution folder should contain only your project folder, named Angband, and the Angband project folder should contain a number of files and directories, including the src and lib folders. These must be directly within the project folder as shown in the image below. In this example, I located my solution folder, named “Angband Master”, next to another solution folder named “Angband 4.2.0”, both under a D:\GitHub folder I use to contain all my GitHub projects. The project folder is highlighted in blue, and the Angband source is located underneath it.



* 1. The VS solution file (.sln filetype) is not checked into the GitHub repository in the location where you need it to be to use it. Copy (or move) it from src\win\vs2019\Angband.sln to your solution directory (Angband Master in the example).
  2. You can change the filename of the solution file to match the name you used on your solution folder; doing so will help you later if you have multiple instances of VS running working on multiple solutions at once.
  3. Move or copy 3 other files, Angband.vcxproj, Angband.vcxproj.filters and Angband.vcxproj.user from the src\win\vs2019 folder to your Angband project directory (Angband Master/Angband). Don’t rename any of these files.
  4. You should now be able to open the solution file (.sln) by using the VS File->Open->Project/Solution menu choice, or you can simply close VS and double-click on the solution (.sln) file to start Visual Studio again and load the solution file.

1. Configuration changes before building
   1. In the tool bar immediately below Visual Studio’s top menu bar there are two adjacent pulldowns. Make sure they are set to ‘Debug’ and ‘x86’ (not x64 – Angband doesn’t build for x64).
   2. The checked-in VS solution and project files contain a number of settings changes that override the default VS settings. These are necessary for building Angband. There shouldn’t be anything extra that you have to set that isn’t already boiled-in to the solution/project files.
2. Code fixes before building
   1. If you are trying to build an older version of Angband, including 4.2.0, you will need to apply some fixes to get it to build and run. These have been fixed in issues #4266 and #4236. The fixes are easy enough to apply by hand if you want to build 4.2.0. No attempt has been made to build versions prior to 4.2.0.
3. F7 to build!
   1. (Hit the Function Key Luke!)
4. Post Build Step
   1. To run, Angband needs to be able to find and load various text files contained under the lib folder. The approach used here is to copy the lib folder to the built executable (versus moving the executable). The reason for that is by default VS supports both Debug and Release configurations, and copying the lib directory to the executable’s location allows multiple configurations to coexist, whereas doing it the other way would cause a collision of the executable files from Release and Debug, and building one configuration would overwrite the executable built by the other configuration.
   2. The provided solution files automate this step completely by doing two xcopy commands after every successful build. You can see the output from these commands in the output window. The first time you build everything will get copied. After that, if nothing has been updated under ‘lib’, then you’ll see “0 files copied” (twice) as the xcopy commands are set up to only copy updated files. If you build successfully and then hit ‘F7’ again, you’ll see that nothing needed to be built but the post-build step will run again, copying 0 files.
   3. If you make changes to lib data, you must use ‘F7’ to “build” your change, which causes xcopy to refresh the copy from lib, so that running Angband will pick up the changes. Often you will change both, and the automation keeps you from having to remember to copy the text files to the executable.
5. Run
   1. Hit ‘F5’ to run what you built under the debugger.